IN-63-CR
217910
12P

Blindness in Designing Intelligent Systems

Peter J. Denning

1 Feb 1988

RIACS Technical Report TR-88.4

NASA Cooperative Agreement Number NCC 2-387

(NASA-CR-184582) BLINDNESS IN DESIGNING INTELLIGENT SYSTEMS (Research Inst. for Advanced Computer Science) 12 p CSCL 09B

N89-25038

Unclas
G3/63 0217910

RIACS

Research Institute for Advanced Computer Science

Blindness in Designing Intelligent Systems

Peter J. Denning

Research Institute for Advanced Computer Science
NASA Ames Research Center

RIACS Technical Report TR-88.4
1 Feb 1988

New investigations of the foundations of artificial intelligence are challenging the hypothesis that problem-solving is the cornerstone of intelligence. New distinctions among three domains of concern for humans -- description, action, and commitment -- have revealed that the design process for programmable machines, such as expert systems, is based on descriptions of actions and induces blindness to nonanalytic action and commitment. Design processes focusing in the domain of description are likely to yield programs like bureaucracies: rigid, obtuse, impersonal, and unable to adapt to changing circumstances. Systems that learn from their past actions, and systems that organize information for interpretation by human experts, are more likely to be successful in areas where expert systems have failed.

This is a preprint of the column *The Science of Computing* for *American Scientist 76*, No 2 (March-April 1988).

Work reported herein was supported in part by Cooperative Agreement NCC 2-387 between the National Aeronautics and Space Administration (NASA) and the Universities Space Research Association (USRA).

Blindness in Designing Intelligent Systems

Peter J. Denning

Research Institute for Advanced Computer Science

1 Feb 1988

Researchers in artificial intelligence have pursued two distinct goals in formulating models of mind: precise descriptions of human thought processes, and tools that apply intelligence. These goals are often mingled: models of thought influence the design of tools, and tools are often touted as themselves models of thought. Most designs for tools are grounded in the assumption that intelligence consists of problem-solving.

Many extravagant claims have been made about the type of software most commonly labeled as an "intelligent tool" -- the expert system (*1*). These claims convey the message that expert systems can succeed in complex and even life-and-death situations where other types of programs have failed, such as in air traffic control, medical diagnosis, the operation of power plants, manufacturing, and weapons systems. I share with a growing number of observers a deep concern at the undue faith being placed in these programs, whose competence and

behavior in untested situations cannot be known.

This concern has stimulated a renaissance of interest in the philosophical foundations of artificial intelligence, a search for the limits of competence of intelligent systems and for a better understanding of how computers might serve people. As scientists, we work within the tradition of logical empiricism, which holds that the world is objectively knowable and that deductions about it can be tested. So steeped in this tradition are we that the conclusions of other, nonempirical areas of study such as ethics or metaphysics -- which are equally rigorous in their reasoning processes -- can easily appear less valid to us because their conclusions are unverifiable. The purpose of the philosophical inquiry is insight into being, from which we may discover new distinctions between humans and machines. What is emerging from this inquiry is an awareness of an expansive blindness that logical empiricism imposes on us, a blindness that prevents us from seeing possible designs that might be more robust than the ones with which we are now working.

How was this blindness introduced? In his 1948 book *Cybernetics*, Norbert Wiener hypothesized that human beings are complicated machines composed of a brain and sensory and motor systems. The human machine is goal-seeking. It is able to detect errors, change course, and adapt its behavior so that achievement of goals is more efficient. Since Wiener's time, two lines of investigation regarding the construction of cybernetic machines have been pursued.

The first is based on the hypothesis that cybernetic machinery is fundamentally symbol-oriented – its operation can be fully described as manipulations of symbols following precise rules without regard to varying interpretations of the symbols. According to this view, intelligent behavior arises from combining symbols in patterns that were not anticipated when the rules were written. Expert systems are products of this line of investigation.

The second line is based on the hypothesis that cybernetic machinery is built from many simple (nonlinear) elements with many interconnections. These "neural networks" store knowledge in their internal states and change states in response to their environments. According to this view, intelligent behavior arises from the collective interactions of many neurons.

There has been abundant debate on the validity of the symbol-processing hypothesis. The argument in favor says that, because any human brain is a system of components that obey the laws of physics and chemistry, the states of the brain can ultimately be described as the solutions to mathematical equations relating (nonlinear) computable functions over the inputs and outputs of neurons, and with sufficient information one could compute a person's next actions. Only two things prevent us from designing a computer program capable of simulating a brain: our limited understanding of neuron functions and their interconnections, and insufficient computing power. In time, say a hundred years, we may have the required understanding and computing power.

The argument against says that humans process everything within a framework of interpretation. Every human conversation has a context, within which what is spoken and heard is interpreted. Humans also make commitments and assume responsibility for their actions. In contrast, rule systems process symbols without regard to their meanings; any attempt to represent background in the rules is bound to fail because much of the background is invisible to us -- it is in our "subconscious". Moreover, designers cannot possibly anticipate the infinity of situations in which the rules might be used, and thus systems are bound to have important blind spots.

If the proponents of the view of human beings as symbol-processing neural networks would regard it as a hypothesis rather than fact, the riptide propelling many of the extravagant claims for expert systems would subside, and a more rational discussion of the problems involved in designing intelligent systems would ensue.

Let's return to the question of intelligent tools. Within logical empiricism, intelligence is naturally modeled by the problem-solving process, which consists of three parts: give a precise statement of the problem and its context, enumerate alternatives for solution, and select an alternative of sufficient payoff and low cost. Expert systems are well suited to this process (*2*).

But is problem-solving an adequate description of intelligence? Douglas Hofstadter argues that it is not (*3*). An essential feature of mind, he says, is the ability to recognize patterns, including patterns in one's own behavior. We have

the ability to recognize that we are caught in a rut and to do something else. Indeed, creative acts occur precisely when we recognize a pattern and intentionally undertake a new behavior. Inherent in creativity is the invention of a new context -- an act outside the problem-solving process.

The work of the philosopher Martin Heidegger, which is described by Terry Winograd and Fernando Flores in *Understanding Computers and Cognition*, sheds more light on this question (*4*). Heidegger distinguished a domain of action from a domain of description. In the domain of action, one reacts to events by bringing know-how into action without conscious thought. One does not have occasion to bring thought to bear on events until a "breakdown" occurs -- an event that interrupts the flow of action toward one's goals. Description is an account of action as it appears to an observer. Something is lost in the translation of action into a description of what happened. The domain of description does not give one full access to the domain of action.

The rules constituting the program of an expert system are descriptions written by an observer of action. The implication of this isn't that expert systems can't compute calls for action fast enough, but that the process of designing a system necessarily creates a blindness that conceals parts of the domain of action from the designer. Consequently, expert systems are bound to miss important cases, to call for unsuitable actions in unanticipated situations, and to behave inappropriately in new contexts. This is why those in the field often call expert systems brittle. Winograd remarks that expert systems can best be com-

pared to bureaucracy in their rigidity, obtuseness, and inability to adapt to changing circumstances (*5*).

It is often argued that this type of blindness can be overcome by interpolating among existing successful expert systems -- for example, by putting all the rules into one database where they can be combined in new, unexpected ways. This argument is challenged by Hofstadter in his analysis of Donald Knuth's assertion that all typefonts can be generated from a few base fonts by appropriate combinations of parameters (*6*). The blindness induced by the problem-solving process cannot be overcome by combining known solutions.

It is important to emphasize that much of the blindness imparted to designs originates in human blind spots and is unavoidable. It has been said that phenomena can be divided into three groups in relation to a person: the known, the known unknown, and the unknown unknown. The known unknown comprises everything the person does not know but knows methods of gaining access to; it is a partial blind spot. The unknown unknown comprises everything a person does not know, and does not know that he does not know. The person does not even possess the language to discuss the unknown unknown; it is a total blind spot. Since a program is an expression in language, it is impossible for a designer to specify procedures for dealing with cases in his own unknown unknown. This is illustrated by an expert system for medical diagnosis designed before the nature of the immune system was understood. Such an expert system would have a blind spot to all immunity-related phenomena, including the asso-

ciated diseases. Medical researchers would be capable of recognizing the blind spot, leading to an explanation of the new phenomena, but the computer would not be.

Some argue that the process of discovering a blind spot and extending the rule set to remove it can be modeled, and that eventually we will have expert systems without blindness that modify their own rules. But this argument supposes that creative human acts ultimately describable by rules -- exactly the hypothesis under question.

Another illustration of blindness involves a designer who is commissioned to create a baseball-playing robot that hits home runs. From the perspective of problem-solving, the designer will begin with the supposition that the "problem" is to compute a realizable trajectory for a bat that intersects with an observed trajectory of a pitched ball. The resulting design will include sensors for detecting the pitched ball's trajectory, fast servomechanisms for swinging the bat, and a powerful trajectory-calculating computer. The designer will look for rules that determine different swings for different possible trajectories of the ball. Now, suppose one also asks professional baseball players what happens at the moment they hit home runs. The answers will be nothing like the designer's suppositions. They are likely to be "the ball was just hanging in the air," or "the moment it left the pitcher's hand, the ball had 'home run' written all over it." These answers occur in the domain of action. The blindness induced by the process of writing a description makes them seem totally useless -- and indeed they are, as

descriptions. And yet the answers suggest a different approach to the design: build a robot that swings at pitched balls, watches its own performance, and amasses a repertoire of experience that leads eventually to its hitting home runs regularly.

In the same way, researchers are making progress with the difficult problem of recognition of continuous speech by designing systems that are shown patterns of encoded speech and learn to recognize those or similar patterns. The new designs are conceived in the domain of action, where learning follows from experience. The older, unsuccessful designs, which looked for rules that mapped speech signals into associated texts, were conceived in the domain of description.

Flores distinguishes a third domain, commitment, which is the source of action. Commitment provides the motivation to deal with breakdowns -- intrusions of the unknown unknown that can be handled only by reformulating the context of a problem so that new, appropriate actions are possible. Understanding the domain of commitment gives us an even deeper appreciation of the source of the brittleness of expert systems, which operate in a fixed context (the one they were designed for) and are incapable of reformulating it. Winograd and Flores say that a good design is a committed attempt to anticipate breakdowns.

Electronic office communication offers a good example of these distinctions. An observer watching a conversation between two people will see a sequence of messages, each consisting of some information followed by an acknowledgement

of receipt. The designs of most electronic mail systems are based on this model of communication. Viewed from the domain of commitment, a typical office conversation includes the intention of the parties to move toward a state of completion in which none expects further communication. The conversation thus consists of requests, offers, counteroffers, promises, and reports of completion. A computer system designed from this perspective will keep track of these components and support the parties as they move toward completion (*4,5*). It is impossible to organize electronic office communications in this way from the domain of description because the observer cannot see the internal states of the parties to the conversation.

So ingrained is the traditional view of problem-solving as the cornerstone of intelligence that new distinctions, such as the domains of description, action, and commitment, seem strange and hard to grasp. And yet coming to grips with them will enable designers to overcome blindness in expert systems and is likely to produce new successes in artificial intelligence.

References

1. P. J. Denning. 1986. "Expert Systems." *American Scientist 74, No. 1. January-February. 18-20.*

2. P. J. Denning. 1986. "Will Machines Ever Think?" *American Scientist 74,* No. 4. July-August. 344-346.

3. D. R. Hofstadter. 1985. "On the Seeming Paradox of Mechanizing Creativity." In *Metamagical Themas.* Basic Books. 526-546.

4. T. A. Winograd and F. Flores. 1986. *Understanding Computers and Cognition: A New Foundation for Design.* Ablex Publishing Co., Norwood, NJ. (Available in paperback from Addison-Wesley. 1987.)

5. T. A. Winograd. 1987. "Thinking Machines: Can There be? Are we?" Report No. STAN-CS-87-1161. June. Computer Science Department, Stanford University, Stanford, CA 94305.

6. D. R. Hofstadter. 1985. "Metafont, Metamathematics, and Metaphysics." In *Metamagical Themas.* Basic Books. 260-296.

7. T. A. Winograd. 1987-88. "A language/action perspective on the design of cooperative work." *Human-Computer Interaction 3*, 1. 3-30.